7주차 결과보고서

전공: 컴퓨터공학과 학년: 2학년 학번: 20211522 이름: 김정환

**1.**

.................

우선 Even Parity bit generator은 아래의 코드로 구현하였다.

텍스트, 스크린샷, 폰트, 라인이(가) 표시된 사진

자동 생성된 설명

이는 주어진 input인 a,b,c,d를 모두 xor한 것으로 구현하였다. 이를 simulation한 결과는 아래와 같다.

스크린샷, 사각형이(가) 표시된 사진

자동 생성된 설명

이에 대한 진리표를 작성하면 아래와 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| In A | In B | In C | In D | Out E |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 |

다음으로 구현 시에 사용한 식으로 간소화할 때 사용한 카르노 맵은 아래와 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | 0 | 1 |
| 01 | 1 | 0 | 1 | 0 |
| 11 | 0 | 1 | 0 | 1 |
| 10 | 1 | 0 | 1 | 0 |

위의 카르노 맵을 정리하면 E = A⊕B⊕C⊕D와 같이 식을 얻을 수 있고, 이를 통해서 위에서 구현한 대로 e = a ^ b ^ c ^ d를 얻을 수 있다.

다음으로 Even Parity bit checker는 다음과 같은 코드로 구현하였다.

텍스트, 폰트, 스크린샷, 영수증이(가) 표시된 사진

자동 생성된 설명

이는 generator와 차이로는 입력으로 하나 더 주어지는 p도 함께 XOR 연산을 하는 점에서 차이가 있다. 이를 simulation한 결과는 아래와 같다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

이에 대한 진리표를 작성하면 아래와 같다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| In A | In B | In C | In C | In P | Out E |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 |

다음으로 구현 시 사용한 식을 간소화할 때 사용한 카르노 맵은 다음과 같다.

우선 P = 0일 때,

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | 0 | 1 |
| 01 | 1 | 0 | 1 | 0 |
| 11 | 0 | 1 | 0 | 1 |
| 10 | 1 | 0 | 1 | 0 |

다음으로 P = 1일 때,

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 1 | 0 | 1 | 0 |
| 01 | 0 | 1 | 0 | 1 |
| 11 | 1 | 0 | 1 | 0 |
| 10 | 0 | 1 | 0 | 1 |

이를 정리하면 E = A⊕B⊕C⊕D⊕P를 얻을 수 있고 이는 Verilog로 나타내면 위와 같이 e = a ^ b ^ c ^ d ^ p로 나타낼 수 있다.

................

**2.**

.......................

Odd Parity bit generator는 아래와 같은 코드로 구현하였다.

텍스트, 폰트, 스크린샷, 영수증이(가) 표시된 사진

자동 생성된 설명

이는 입력으로 주어진 a, b, c, d를 모두 XNOR한 것으로 구현하였다. 이를 simulation한 결과는 아래와 같다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

이에 대한 진리표를 작성하면 아래와 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| In A | In B | In C | In D | Out E |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

다음으로 구현한 식을 간소화할 때 사용한 카르노 맵은 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 1 | 0 | 1 | 0 |
| 01 | 0 | 1 | 0 | 1 |
| 11 | 1 | 0 | 1 | 0 |
| 10 | 0 | 1 | 0 | 1 |

이를 정리하면 E = (A⊕B⊕C⊕D)’와 같은 식을 얻을 수 있고, 이는 구현한 식과 동일하게 나오는 것을 볼 수 있다.

다음으로 Odd Parity bit checker는 아래와 같은 코드로 구현하였다.

텍스트, 폰트, 스크린샷, 친필이(가) 표시된 사진

자동 생성된 설명

이는 generator에서 XNOR 연산을 추가된 입력인 p와 함께 하는 방식으로 구현하였다. 이를 simulation한 결과는 아래와 같다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

이에 대한 진리표를 작성하면 아래와 같다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| In A | In B | In C | In C | In P | Out E |
| 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 |

다음으로 구현한 식을 간소화할 때 사용한 카르노 맵은 다음과 같다.

P = 0일 때

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 1 | 0 | 1 | 0 |
| 01 | 0 | 1 | 0 | 1 |
| 11 | 1 | 0 | 1 | 0 |
| 10 | 0 | 1 | 0 | 1 |

P = 1일 때

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | 0 | 1 |
| 01 | 1 | 0 | 1 | 0 |
| 11 | 0 | 1 | 0 | 1 |
| 10 | 1 | 0 | 1 | 0 |

이를 정리하면 E = (A⊕B⊕C⊕D⊕P)’가 나오는 것을 확인할 수 있고 이는 위에서 구현한 식과 동일하다.

.........................

**3.**

.......................

2-bit binary comparator를 구현한 코드는 아래와 같다.

텍스트, 폰트, 스크린샷, 대수학이(가) 표시된 사진

자동 생성된 설명

각각 f1이 A > B, f2가 A=B, f3가 A<B인 경우를 출력한다. 이를 simulation한 결과는 아래와 같다.

스크린샷, 사각형이(가) 표시된 사진

자동 생성된 설명

각 결과에 대한 진리표는 아래와 같다.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| In A | In B | In C | In D | Out F1 | Out F2 | Out F3 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 | 1 | 0 |

위에서 구현한 식으로 간소화하는데 사용한 카르노 맵은 다음과 같다.

우선 F1에 대한 카르노 맵은 아래와 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | 1 | 1 |
| 01 | 0 | 0 | 1 | 1 |
| 11 | 0 | 0 | 0 | 0 |
| 10 | 0 | 0 | 1 | 0 |

이를 정리하면 F1 = AC’ + BC’D’ + ABD’의 식을 얻을 수 있다.

다음으로 F2에 대한 카르노 맵은 아래와 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 1 | 0 | 0 | 0 |
| 01 | 0 | 1 | 0 | 0 |
| 11 | 0 | 0 | 1 | 0 |
| 10 | 0 | 0 | 0 | 1 |

이를 정리하면 F2 = A’B’C’D’ + A’BC’D + AB’CD’ + ABCD = (A⊕C)’(B⊕D)’로 정리할 수 있다.

다음으로 F3에 대한 카르노 맵은 아래와 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 0 | 0 |
| 01 | 1 | 0 | 0 | 0 |
| 11 | 1 | 1 | 0 | 1 |
| 10 | 1 | 1 | 0 | 0 |

이를 정리하면 F3 = A’C + A’B’D + B’CD로 정리할 수 있고, F1, F2, F3 모두 구현 시 사용한 식이 나오는 것을 볼 수 있다.

.........................

**4.**

.......................

본 실험을 통해 Even Parity bit generator 및 checker, Odd Parity bit generator 및 checker 그리고 2-bit binary comparator를 Verilog로 구현하고, FPGA에서의 작동을 확인하였다. 4변수 카르노 맵과 5변수 카르노 맵을 활용하여 논리식을 간소화하였고, 간소화 결과가 옮게 나오는 것을 simulation과 FPGA로 확인했다. Even Parity bit의 경우에는 Parity bit 포함 1의 개수가 짝수여야 하므로 generator에서는 1의 개수가 홀수일 때 출력인 Parity bit가 1이 되도록, checker에서는 총 5 bit의 입력에서 1의 개수가 홀수일 때 에러인 1을 출력하도록 구현했다. 이는 모두 XOR Gate로 간소화되는 것을 확인했다. Odd Parity bit의 경우에는 Even과 반대의 결과여야 하므로 Even Parity bit에서 구현한 결과의 NOT임을 추측할 수 있다. 이를 식을 간소화할 때 똑같이 나오는 점을 확인할 수 있었다. 마지막으로 2-bit binary comparator는 각각의 결과들을 카르노 맵으로 정리하여 3가지 식을 정리했다. 이때 A>B와 A<B의 경우에는 서로 NOT을 씌운 것과 동일한 점도 식에서도 진리표에서도 모두 확인할 수 있다.

.........................

**5.**

.......................

이외에 존재하는 조합논리회로의 종류 중 디코더가 존재한다. 이는 N bit의 이진수를 입력받아 2 \*\* N 개의 출력들 중 하나로 바꿔주는 회로이다. 2 bit를 입력 받아 4 bit로 출력하는 디코더의 예시를 들면, 해당 디코더의 진리표는 아래와 같다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| In A | In B | Out D0 | Out D1 | Out D2 | Out D3 |
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 |

이러한 방식으로 나타내고, 각각의 출력에 대한 식은 아래와 같다.

D0 = A’B’

D1 = A’B

D2 = AB’

D3 = AB

이외에도 3 bit의 입력을 받아 8 bit의 출력을 나타내는 디코더나 신호를 받는 디코더, 4 bit를 입력받아 16 bit의 출력을 나타내는 경우 등으로 다양한 디코더가 존재한다.

.........................